ANTI-ALIASING LINE PIXEL COVERAGE CALCULATION USING PROGRAMMABLE SHADER

FIELD OF THE INVENTION

[0001] The present invention relates generally to computer graphics and more particularly to the drawing of lines to minimize or reduce aliasing effects from a limited resolution output device.

DESCRIPTION OF THE RELATED ART

[0002] Drawing lines is a fundamental operation in computer graphics systems. Commonly, these computer graphics systems employ displays that have a certain limited number of picture elements (pixels) which determine the resolution of the display. The limited resolution of the display has an effect on the drawing of lines, however, because each pixel of the display is in effect a spatial sampling point for the line to be drawn. It is well-known that when there are too few sampling points for the spatial frequency inherent in an object being represented, including a line, then the limited resolution of the display may either poorly represent, misrepresent, or completely miss the object. This phenomenon is called aliasing and occurs not only for spatially sampled objects but also for temporally sampled events.

[0003] The result of aliasing when lines are drawn on a computer graphics display is that for some lines, usually lines at certain angles, rather than horizontal or vertical lines, the line appears ragged on its edges. This ragged appearance of the line is unpleasant to the user and makes it difficult to render clean looking drawings on the display or even a resolution limited printing device. Relatively thin lines make the lines look worse. One can increase the resolution of the graphics display or other output device in an effort to reduce this problem. However, the increased number of pixels requires a much greater amount of processing and much greater amount of memory for storing the pixels. Therefore, it has become desirable to seek means for reducing or eliminating the ragged appearance of the drawn lines by a variety of anti-aliasing techniques, among which are pre-filtering, supersampling and postfiltering.

[0004] Pre-filtering involves computing the fractional coverage of a pixel by a non-zero thickness line and using that fraction to decide what the pixel intensity should be. In order to

determine the fractional coverage of a pixel, an algorithm used to draw the line must determine the pixels to be included in rendering the line on the display.

[0005] One way to determine whether a pixel should be included in the rendering of a line, is to consider a pixel as a unit-size rectangle having a center and determine whether or not that center is within the bounds of the line. FIG. 1 shows the pixels 12-54 involved in drawing a line segment 10 in accordance with this pixel-center rule. As is clearly observable, the pixels 12-54 included in the line segment 10 shown give the segment a somewhat jagged appearance. As mentioned above, it is desirable to improve on the appearance of such a line segment without increasing the number of pixels in the output device.

BRIEF SUMMARY OF THE INVENTION

[0006] The present invention is directed towards realizing the above-mentioned improvement. A method in accordance with the present invention includes a method in a computer graphics system for rendering on an output device, having a finite number of pixels, a non-zero thickness line segment with reduced aliasing. The method includes expanding an edge of the line segment touching but not covering a pixel center of the line segment to be rendered on the output device so that the expanded line segment covers the center of the pixel touched. Then, using the pixel centers, the pixels to be included in the expanded line segment are determined, where the line segment being distinguishable from a background over which said line segment is rendered by having a shade or color different from a shade or color of the background. For each pixel that is included in said non-zero thickness expanded line segment, the area of the pixel partially- or fully-covered by the line segment is determined and based on the area of the pixel covered, a shading or color value for the pixel is determined by interpolating between the shade of the line segment and the shade or color of the background.

[0007] One advantage of the present invention is that aliasing is of the line segment is reduced or minimized, thereby permitting more accurate and better appearing lines rendered on the output device.

[0008] Another advantage of the present invention is that the computational cost of performing the anti-aliasing of the present invention is reduced.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] These and other features, aspects and advantages of the present invention will become better understood with regard to the following description, appended claims, and accompanying drawings where:

FIG. 1 shows the appearance of a line on a limited resolution output device when a non-zero thickness line is drawn in accordance with a conventional pixel-center rule;

FIG. 2 shows the appearance of a line on a limited resolution output device when a non-zero thickness line is expanded to include pixels that are touched but whose centers are not within the borders of the original line;

FIG. 3 shows a non-zero thickness line covering pixels in the output device such that only one edge of the line traverses any one pixel;

FIG. 4A. shows a non-zero thickness line covering pixels in the output device such that two or three edges of the line may traverse at least one of the pixels;

FIG. 4B shows an magnified version of the case where three edges traverse one of the pixels;

FIG. 5 shows a non-zero thickness line covering pixels in the output device such that four edges of a line may be present in at least one of the pixels;

FIG. 6 shows an enlarged area of the output device depicting a set of pixels, the movement of the line boundary and the primitive triangles and parallelogram used to compute pixel coverage in accordance with the present invention;

FIG. 7 shows a block diagram of the graphics pipeline used to implement the above computations;

FIG. 8 is a flow chart showing a method in accordance with the present invention of rendering a line segment on an output device with reduced aliasing;

FIG. 9 is a flow chart showing the steps for carrying out the adjustment of the edge of the line segment;

FIG. 10 is a flow chart showing the steps for determining the pixels to be included in the rendering of the expanded line segment;

FIG. 11 is a flow chart showing the steps for determining the area of a pixel covered by the expanded line segment;

FIG. 12A is a flow chart showing the steps for determining the area covered by an edge traversing a pixel;

FIG. 12B is a flow chart showing the steps of an alternative method for determining the area covered by an edge traversing a pixel; and

FIG. 13 is a flow chart showing the step for determining the shading or color value of a pixel after determining the area of the pixel covered by the line segment.

DETAILED DESCRIPTION OF THE INVENTION

[00010] In the discussion that follows, it is assumed that the equation of the line, $ax + by + c = 0$, is known and that a coordinate system is imposed on the output device with the x-direction increasing to the right and the y-direction increasing downward. Also, for purposes of the present discussion, the line is assumed to be sloping upwards to the right, such that $a \geq b \geq 0$. It is also convenient to define certain parameters of the line. First, the slope m of the line is defined as $|a|/|b|$ and the slope-factor sf for the line is defined as $m/(m+1)$. Using the latter definition is helpful because as the slope ranges from 0 to ∞, the slope factor ranges from 0 to 1. The slope m of the line may be expressed as a function of the slope factor, $m=sf/(1-sf)$. Second, a parameter, $p = sf \cdot \Delta x$, which is the product of the slope factor sf of the edge equation of the line segment and an x-directed displacement, Δx, will prove useful.

[00011] In accordance with the present invention, the pixels to be included in the rendering of a non-zero thickness line must be determined. Given a pixel P with coordinates (x_0, y_0) and one line edge relation $ax + by + c \geq 0$, to determine whether a pixel P is inside the edge, the line segment edge relation $ax + by + c \geq 0$ is evaluated at the pixel coordinate (x_0, y_0). If the result is greater than or equal to zero, the pixel is inside the edge. If the pixel is inside all of the four edges of line segment then it is to be included in the rendering of the line.

[00012] FIG. 2 shows the appearance of a line segment 80 on a limited resolution output device when a non-zero thickness line is rendered using the centers of the pixels to decide whether the pixel is to be included. To improve the appearance of the line segment 80 shown in FIG. 1, it is desirable that those pixels 56-78 that are touched by the non-zero thickness line but whose centers are not included within the bounds of line segment be included. As described

above, these pixels 56-78 are not included in a line drawing algorithm that bases the inclusion of the pixel on whether or not the pixel center is covered by the non-zero thickness line. To include these pixels in the rendering of the line segment, the boundaries of the line segment are expanded, in accordance with the present invention, to include the centers of pixels that are touched by the edge of the line segment. The expansion of the line segment's edge can be interpreted as a change Δc in the c parameter in the equation of the line edge.

[00013] Referring to FIG. 2 and FIG. 6, and taking one edge as example, suppose that the left edge of the original line 80 in FIG. 2 just touches the right bottom corner of the pixel 82, thereby partially covering the pixel. In accordance with the present invention, the edge of the line segment is expanded (shown as expanded line segment 84 in FIG. 2) to the left to include the center of the just barely touched pixel 82. In the x-direction, this expansion amounts to a displacement 86 in the x-coordinate of the line segment edge from x' to x_0 in FIG. 6. The displacement $x' - x_0 = \Delta x = \frac{b}{2a} + \frac{1}{2} = \frac{a+b}{2a}$, where the first term is the x-directed distance from x' to the pixel boundary and the second term is the x-directed distance from the pixel boundary to the pixel center x_0. This is the amount by which the c parameter must be adjusted to move the edge of the line over to include the center of a partially covered pixel. Note that the slope of the line is not being altered, only the y-intercept of the edge of the line. Because $x' - x_0 = \frac{a+b}{2a}$, the new line edge relation becomes $ax_0 + by_0 + c + \frac{a+b}{2} \geq 0$.

[00014] To determine whether a pixel is included in the rendering of the line now translates into evaluating the edge relation of the line $ax + by + c + \frac{a+b}{2} \geq 0$ at (x_0, y_0). Here it is assumed that a ≥ 0, b ≥ 0, for all other cases it is easy to prove that the same amount of adjustment $\Delta c = \frac{|a|+|b|}{2}$ applies to the other edges as well.

[00015] After the edges of the non-zero thickness line have been adjusted, it remains to compute the fractional area f of the included pixels covered by the line, assuming that the area of

a pixel is unity. With the fractional area known, the shading or color of the pixel is then determined, in accordance with the invention, by linear interpolation between the shading or color C_{LINE} of the line and the shading or color of the background C_{BG}, by $f*C_{LINE} + (1-f)*C_{BG}$.

[00016] To determine the fractional area *f* of coverage of a pixel, several cases must be considered. These cases include a single edge 90 in FIG. 3 of the line traversing a pixel (case I), two parallel edges 92 94 in FIGs. 4A, 4B of the line traversing a pixel (case II), two edges 92, 96 in FIGs. 4A, 4B of the line, orthogonal to each other, traversing the pixel (case III), and three edges 92, 94, 96 in FIG. 4B (case IV) or four edges 92, 94, 96, 98 in FIG. 5 traversing a pixel (case V). If *c0* is the area not covered by the line when one edge traverses a pixel, then 1-*c0* is the area covered in case I. If *c0* and *c1* are the areas not covered by the line when two parallel edges, respectively traverse the pixel, then the area covered is (1-(*c0*+*c1*)). If *c0* and *c2* are the areas not covered by the line when two orthogonal edges, respectively traverse the pixel, then the area covered can be approximated as (1-*c0*)*(1-*c2*). At the pixels near the end of the line segment, it may be true that three edges of the line traverse a pixel, such as is shown in FIG. 4B. In this case, if *c2* is the area not covered by the third edge and *c0* and *c1* are defined as in cases I and II, then the area covered can be approximated as (1-(*c0*+*c1*))*(1-*c2*). Finally, in the rare case, FIG. 5, when four edges of a line traverse a pixel, and *c3* is defined as the area not covered by the fourth edge, and the other edges are as defined in cases I, II and III, then the area covered can be approximated as (1-(*c0*+*c1*))*(1-(*c2*+*c3*)). In cases III, IV and V, the formulas for coverage are an approximation that is sufficiently accurate for the purposes involved. It is desirable to limit the range of areas *c0*, *c1*, *c2*, and *c3* to the interval of [0,1]. This allows the formulas to be used on all four edges without an SIMD processor having to determine whether the edge is alongside or perpendicular to line direction and without the need for branching or testing of conditions. A simplification of the above in one embodiment includes assuming that the coverage calculation for the two edges perpendicular to the line direction is skipped and the corresponding areas are zero when performance is more important than the quality of appearance of the line.

[00017] From the above, it is clear that if the coverage of a pixel by a single line edge can be determined, the coverage for the others cases can be derived, as they are just similar applications of the coverage of a pixel by one line edge.

[00018] The determination of the coverage of a pixel by a line when a single line edge traverses the pixel requires that two cases be considered. The first case is when the edge traverses the pixel so as to cover only a triangular portion 100, in FIG. 6, of the pixel. The second case to consider is when the edge traverses the pixel so as to cover both a triangular portion 100 and a parallelogram portion 102 in FIG. 6.

[00019] When the line edge carves out a triangular portion of the pixel, the area A_0 covered by the line is given by $\frac{1}{2}\Delta x \cdot \Delta y$, where Δx is the base of the triangle and Δy is the height of the triangle 100, as shown in FIG. 6. Of course, Δx and Δy are related by the edge equation of the line. In particular, if $\Delta y = m\,\Delta x$, and $\Delta x = p/sf$, then the equation for the area becomes

$$A_0 = \frac{1}{2} m \cdot (p/sf)^2, \text{ or}$$

$$A_0 = \frac{1}{2} p^2 \cdot (1-sf)^{-1} \cdot sf^{-1},$$

which is a form particularly well suited for computation in a graphics pipeline described below.

[00020] When the edge of the non-zero thickness line carves out both a triangle 100 and a parallelogram 102, the area of the parallelogram must be determined and added to the maximum area of the triangle to determine the area of coverage. The maximum area of the triangle 100, assuming that $1/2 \le sf \le 1$, occurs when $p = (1-sf)$, and is

$$\max(A_0) = \frac{1}{2m} = \frac{1}{2}(1-sf) \cdot sf^{-1},$$

where it is assumed that the area of a whole pixel is unity, each side of the pixel being unity. The area A_1 of the parallelogram 102 is

$$\Delta x_{\|} = \Delta x_{TOT} - \frac{1}{m} = p \cdot sf^{-1} + (1 - sf^{-1}),$$

because its height is unity. The maximum area $\max(A_1)$ of the parallelogram 102 is 1-1/m. The maximum area of both is

$$\max(A_0) + \max(A_1) = 1 - (1+sf)(2sf)^{-1}.$$

Alternatively, the above formulas may be expressed as a conditional function of p:

$$\text{if } 0 \le p \le (1-sf), \text{ then } A(p) = \frac{1}{2} p^2 \cdot (1-sf)^{-1} \cdot sf^{-1})$$

$$\text{if } (1-sf) < p \le 1, \text{ then } A(p) = \frac{1}{2}(1-sf)^2 \cdot (1-sf)^{-1} \cdot sf^{-1} + p \cdot sf^{-1} + (1-sf^{-1}).$$

[00021] The remaining area of the pixel is just the remaining triangle A_2 104, whose area is again $\frac{1}{2m}$. Thus, if the area covered is greater than $\max(A_0) + \max(A_1)$, the area of coverage is obtained by computing the triangular area,

$$A_2 = \frac{1}{2} r^2 \cdot (1-sf)^{-1} \cdot sf^{-1}$$

and by subtracting it from 1, to give $1 - A_2$,

where the distance Δx in the parameter r, is measured from the left side of the pixel, thus $r = 1 - p$. Therefore, the area of coverage is then,

$$A_3 = (1 - A_2) = 1 - \frac{1}{2}(1-p)^2 \cdot (1-sf)^{-1} \cdot sf^{-1}.$$

The conditional formulas for the case when $0 \le sf \le 1/2$ are:

$$\text{if } 0 \le p \le sf, \text{ then } A(p) = \frac{1}{2} p^2 \cdot (1-sf)^{-1} \cdot sf^{-1}), \text{ and}$$

$$\text{if } sf < p \le 1, \text{ then } A(p) = \frac{1}{2} sf^2 \cdot (1-sf)^{-1} \cdot sf^{-1} + p \cdot sf^{-1} + (1-sf^{-1}).$$

As can be observed, if sf is greater than 1/2 then the limits on p in the conditional are (1-sf) and if sf is less than 1/2, the limits on p are sf. Therefore, the limits on p are always the min(sf, 1-sf). This suggest a way to combine the conditional formulas for both cases:

$$\text{if } 0 \le p \le \min(sf, 1-sf), \text{ then } A(p = \min(sf, 1-sf)) = \frac{1}{2} p^2 \cdot (1-sf)^{-1} \cdot sf^{-1}), \text{ and}$$

$$\text{if } \min(sf, 1-sf) < p \le 1, \text{ then } A(p = \min(sf, 1-sf)) = \frac{1}{2} sf^2 \cdot (1-sf)^{-1} \cdot sf^{-1} + p \cdot sf^{-1} + (1-sf^{-1}).$$

[00022] Referring to FIG. 6, in one embodiment calculating a displacement Δx in the x-direction into a pixel when an edge of a line segment passes through coordinates (x_c, y_c) in the pixel 82, includes evaluating the edge relation $ax + by + c \geq 0$ at (x', y'),

$$P_{(x',y')} = ax' + by' + c,$$

with $y' = y_c$, and with $x' = x_c + \Delta x$. Therefore,

$$P_{(x',y')} = ax_c + by_c + c + a\Delta x.$$

Because the edge of the line traversing the pixel actually passes through coordinates (x_c, y_c) in the pixel,

$$ax_c + by_c + c = 0.$$

Therefore, the edge relation being evaluated reduces to

$$P_{(x',y')} = a\Delta x.$$

[00023] The edge relation $ax + by + c \geq 0$ evaluated at (x', y') can also be expressed in terms of the coordinates of the center (x_0, y_0) of the pixel as $P_{(x',y')} = ax_0 + by_0 + c + \frac{a+b}{2}$, where the equality $x' = x_0 + (a+b)/2a$ has been substituted into the above equation, the y values being the same. Therefore, setting the two versions of the edge relation equal to each other, gives

$$P_{(x',y')} = a\Delta x = ax_0 + by_0 + c + \frac{a+b}{2}, \text{ which can be used to solve for } \Delta x,$$

$$\Delta x = \frac{ax_0 + by_0 + c + (a+b)/2}{a},$$

In this case, $a > 0$, $b >= 0$ is assumed. Thus, the displacement Δx for a given pixel is a function of constants and the parameters a, b and c.

[00024] When the edge traverses (touches) the pixel, the displacement Δx is in the range of [0, 1+b/a], and $a\,\Delta x$ is in the range of [0, a+b], assuming that $a > 0$, $b \geq 0$. It is desirable to scale -($P_{(x',y')} = a\Delta x$) to be in the range of [0,1], by multiplying the above equation by (1 /(a+b)), Thus,

$$p = P_{scaled} = a\Delta x/(a+b) = ax_0/(a+b) + by_0/(a+b) + c/(a+b) + 1/2.$$

This makes it easier to use existing interpolation logic to perform the calculation. For all other cases,

$$p = ax_0/(|a|+|b|) + by_0/(|a|+|b|) + c/(|a|+|b|) + 1/2,$$

which expressed in terms of *sf* becomes:

$$p = sf \cdot x_0 + (1-sf) \cdot y_0 + sf \cdot c/a + 1/2$$

[00025] FIG. 7 shows a block diagram of the graphics pipeline used to carry out the above computations. An interpolator is used to compute the parameter $p = sf \cdot \Delta x$, which is a function of the x-distance spanned by the area covered by the line edge, based on the general formula $P = P_s + P_x \cdot x + P_y \cdot y$, which the interpolator implements to compute the shading of triangles. Matching like terms requires that $P_x = sf$, $P_y = (1-sf)$, and $P_s = sf\frac{c}{a} + 1/2$, in the interpolator formula.

[00026] If the computed parameter *p* is negative, it is clamped to zero, and the pixel is not covered or touched by the line edge. If the result is in the range of [0,1], the pixel is touched by the edge and the result is used to calculate the area of coverage in the pixel shader. If the result is greater than 1, it is clamped to 1, and the pixel is 100% covered by the line edge.

[00027] The interpolator is a fixed function SIMD processor capable of operating on multiple data items with the same instruction. An instruction-based shader is then used to compute the coverage area based on the *x*-spanned distance, in accordance with the above formulas.

[00028] FIG. 8 is a flow chart showing a method in accordance with the present invention of rendering a line segment on an output device with reduced aliasing. There are two loops in the method. The first loop 202 expands the edges of the line segment to include pixels that are touched but do not have their centers covered by the line segment. The second loop 204 determines the shading or coloring of each pixel that is included in the expanded line segment.

[00029] The first loop 202 includes the step 206 of adjusting the edge of the line segment to traverse the center of a pixel partially covered and the step 208 of determining, based on the

pixel centers, which pixels of the output device are included in the non-zero thickness line segment. The second loop includes the step 210 of determining the area covered of each pixel included in the expanded line segment, and the step 212 of determining the shading or color of the pixel based on the area.

[00030] FIG. 9 is a flow chart showing the steps for carrying out the adjustment of the edge of the line segment, step 206. In FIG. 9, label A is the entry and label A′ is the return. First, the edge relation for the selected edge is obtained in step 220 and the edge is adjusted by adding, in step 222, a quantity $(|a| + |b|)/ 2$ to the edge relation. This effectively expands the edge to include pixels touched but with centers not covered.

[00031] FIG. 10 is a flow chart showing the steps for determining the pixels to be included in the rendering of the expanded line segment, step 208. First, a pixel is selected, in step 224 and the coordinates of its center are determined. Next, in step 226, using the coordinates the edge relation of the expanded line is evaluated. If the result is true, as determined in step 228, meaning that the evaluated expression is greater than or equal to zero, the pixel is determined to be included, in step 230, in the line segment. Otherwise, in step 232, it is not included. This continues for all of the selected pixels, as determined by step 234.

[00032] FIG. 11 is a flow chart showing the steps for determining the area of a pixel covered by the original line segment, step 210. First, all four parameters, c0, c1, c2 and c3 are computed in step 242. Then, based on the number of edges traversing the pixel, (i.e., depending on which parameters are non-zero) as determined in step 248, the area covered is computed. In step 240, the area covered, when one edge traverses the pixel, is computed. In step 244, the area covered, when two parallel edges traverse the pixel, is computed. In step, 246, the area covered when one edge and another orthogonal edge traverse the pixel, is computed. In step 250, the area covered when three edges traverse the pixel, is computed. In step 258, the area covered when four edges traverse the pixel, is computed. Each of the steps 240, 244, 246, 250, and 258 relies on the steps of FIG. 12 to compute the relevant area.

[00033] FIG. 12A is a flow chart showing the steps for determining the area covered by an edge traversing a pixel. First, p is calculated in pixel calculation block (fixed function interpolator), then in programmable pixel shader, the pixel coverage is calculated based on the

four parameters c0, c1, c2, c3. Each of the steps in FIG. 11 uses the steps of FIG. 12 to determine the area covered by the edge in question. First, a slope factor *sf* is computed in step 262, followed by a computation for the parameter *p* in step 264. The parameter p computation uses the formula $p = ax_0/(|a|+|b|) + by_0/(|a|+|b|) + c/(|a|+|b|) + 1/2$, derived above. Next, if as determined in step 266, the area is less than the maximum triangular area, the triangular area is computed, in step 268, using the parameter *p* and the slope factor *sf*. If the area is greater than the maximum triangular area but less than the sum of the maximum triangular area and maximum parallelogram area, as determined in step 270, or alternatively if 0<=p<(1-sf), then the parallelogram area is computed in step 274 and added to the maximum triangular area in step 276, which is computed in step 272. If the area is greater than the maximum triangular area and maximum parallelogram area, as determined in step 270, then the remaining triangular area is computed and subtracted from one , in step 278. The result is the area of a pixel covered by an edge traversing the pixel.

[00034] FIG. 12B is a flow chart showing an alternative method for computing the area of a pixel covered by an edge traversing the pixel. In step 261, the slope factor sf is computed and in step 263, the p parameter is computed. In step 265, a determination is made as to whether the *p* parameter is between 0 and min(*sf, 1-sf*). If so, then the area is computed according to step 267. If not, i.e., *p* is between min(*sf, 1-sf*) and 1 (as determined in step 269), then the area is computed according to step 271.

[00035] FIG. 13 is a flow chart showing the step for determining the shading or color value of a pixel after determining the area of the pixel covered by the line segment. In step 282, the fractional area covered and the colors or shadings of the line and background are obtained. In step 284, a linear interpolation calculation is performed to obtain the shading or color of the pixel.

[00036] Although the present invention has been described in considerable detail with reference to certain preferred versions thereof, other versions are possible. Therefore, the spirit and scope of the appended claims should not be limited to the description of the preferred versions contained herein.